Hierarchical Extraction of Iso-Surfaces with Semi-Regular Meshes

Kai Hormann[*] Ulf Labsik[†] Martin Meister[‡] Günther Greiner[§]

Computer Graphics Group, University of Erlangen

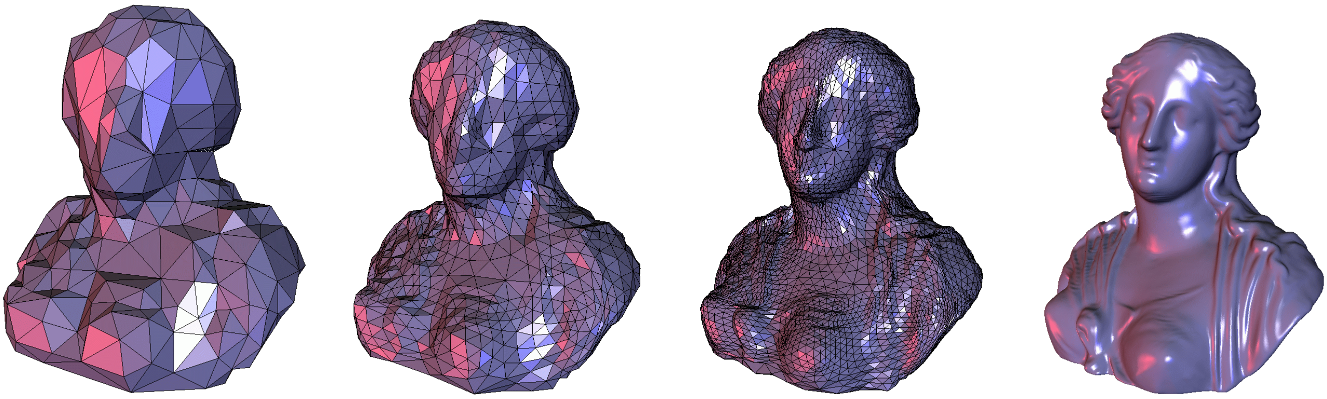

Figure 1: First three levels and final result of our hierarchical iso-surface extraction algorithm.

Abstract

In this paper we present a novel approach to iso-surface extraction which is based on a multiresolution volume data representation and hierarchically approximates the iso-surface with a semi-regular mesh. After having generated a hierarchy of volumes, we extract the iso-surface from the coarsest resolution with a standard Marching Cubes algorithm, apply a simple mesh decimation strategy to improve the shape of the triangles, and use the result as a base mesh. Then we iteratively fit the mesh to the iso-surface at the finer volume levels, thereby subdividing it adaptively in order to be able to correctly reconstruct local features. We also take care of generating an even vertex distribution over the iso-surface so that the final result consists of triangles with good aspect ratio. The advantage of this approach as opposed to the standard method of extracting the iso-surface from the finest resolution with Marching Cubes is that it generates a mesh with subdivision connectivity which can be utilized by several multiresolution algorithms. As an application of our method we show how to reconstruct the surface of archaeological items.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Algorithms

Keywords: Multi Resolution Models, Geometric and Topologic Representations, Reverse Engineering

1 Introduction

Iso-surface extraction from volume data as obtained, for example, from CT scans is a standard technique in scientific visualization. Typically, such iso-surfaces are represented as a triangle mesh and the Marching Cubes algorithm (MC) is commonly used for constructing it. The main drawback of that method is that it produces many small and badly shaped triangles which require improving the mesh with decimation, smoothing, or remeshing. These post processing algorithms can be very time and memory consuming, especially if the meshes are large. And with the resolution of today's CT scanners, the output mesh of MC can easily consist of millions of triangles.

We therefore propose to down-scale the volume data set and create a hierarchy of volumes by iteratively applying a dilation operator as described in Section 3. Then we use MC to extract the iso-surface on the coarsest resolution and fit the mesh to the iso-surfaces at the finer levels of the volume hierarchy later. Since the number of triangles in the extracted mesh depends quadratically on the resolution of the volume, performing MC on the coarsest level yields a mesh with low complexity which can be optimized efficiently. We present, in fact, a simple strategy for improving the MC mesh by removing short edges so as to obtain a base mesh with few and well-shaped triangles.

Once this base mesh is constructed, we use it as an initial guess for approximating the iso-surface on the next finer volume level and iterate this fitting process until we arrive at an iso-surface reconstruction with respect to the original data. Our fitting procedure is discussed in Section 4 and takes three aspects into account. Firstly, the vertices of the mesh need to be projected onto the iso-surface as we want to sample that surface. Secondly, a relaxation operator is required to evenly distribute the sample points over the surface and to ensure well-shaped triangles in the final mesh. Thirdly, we adaptively subdivide the mesh in order to approximate the iso-surface within a user-specified accuracy and to capture local detail. In this way we finally obtain a semi-regular mesh with a hierarchical structure that can be utilized by many multiresolution algorithms such as level-of-detail rendering [3], progressive transmission [10, 14], multiresolution editing [31], and wavelet decomposition [22, 18].

As an application of the method, our special interest lies in archaeological objects like the one in Figure 1, amphoras, and vases. We present some results of our algorithm in Section 5 and finally conclude in Section 6.

[*] hormann@cs.fau.de

[†] labsik@cs.fau.de

[‡] mnmeiste@immd9.cs.fau.de

[§] greiner@cs.fau.de

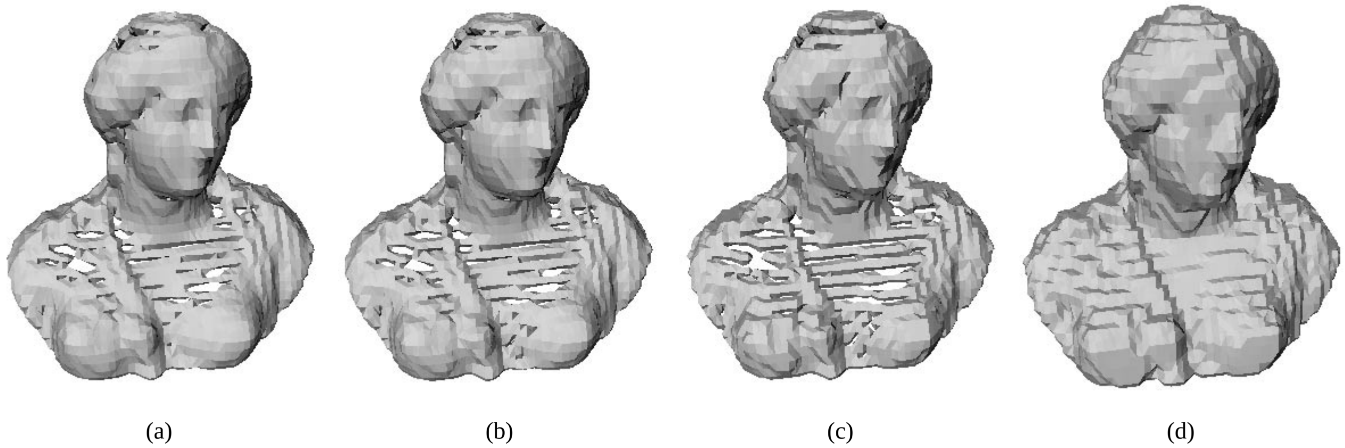

Figure 2: Iso-surface $M^2(900)$ using box filter (a), Gauß filter (b), median filter (c), and dilation (d) to compute f^2.

2 Related Work

The standard approach for the extraction of iso-surfaces from volume data is the well known Marching Cubes (MC) algorithm [17]. The algorithm walks through all cells of a regular hexahedral grid and computes the iso-surface for each cell independently. In order to avoid ambiguities of MC, several modifications were proposed [20, 19] and an extension to reconstruct surfaces with sharp features from distances volumes was presented by Kobbelt et al. [13]. Several algorithms [6, 27, 29] were proposed using adaptive hierarchies of a volume dataset to extract iso-surfaces.

The task of converting an arbitrarily triangulated mesh into a semi-regular mesh is called remeshing. In the approach of Eck et al. [5], vertices are distributed over the given triangulation and a base mesh is constructed by growing geodesic Voronoi tiles around the vertices. A parameterization of the given triangulation within the base triangles is computed by using harmonic maps which minimize the local distortion. The remesh is then determined by uniformly subdividing each base triangle and mapping the vertices into 3-space using the parameterization. Lee et al. [15] construct the base mesh by mesh reduction based on edge collapses and incrementally compute a parameterization of the original triangulation within the triangles of the remaining mesh. This process leads to a locally smooth parameterization. In order to achieve a global smoothness the dyadic points are moved by a variant of Loop's subdivision scheme and mapped into 3-space. Kobbelt et al. [12] describe a shrink-wrapping approach for remeshing. The idea is to place a semi-regular mesh around the original surface. Analogously to the physical shrink-wrapping by exhausting the air between both surfaces the semi-regular mesh is shrunk onto the surface. In addition, a relaxation force is used to distribute the vertices uniformly over the surface.

The direct extraction of semi-regular meshes from volume data is addressed by several papers. Bertram et al. [2] use MC to extract an initial iso-surface which is coarsened by a mesh simplification algorithm based on [7]. Then they use a modified shrink-wrapping approach to compute their final semi-regular mesh based on a quadrilateral subdivision scheme. A method for directly extracting a coarse base mesh from the volume was presented by Wood et al. [30]. They compute contours of the surface from the volume data and connect them topologically correct. The final semi-regular mesh is constructed by using a multiscale force-based solver with an external force moving the vertices to the iso-surface and an internal force relaxing the vertices of the mesh.

3 Base Mesh Construction

In order to efficiently create a base mesh with few triangles, we propose to run a marching cubes algorithm on a coarse volume which is computed by down-sampling the given data. As the number of triangles generated by marching cubes depends quadratically on the number of voxels in each dimension, scaling down the volume by a factor of n reduces the complexity of the extracted mesh by n^2.

Suppose the volume data to be represented as a discrete gray value function $f^0 : G^0 \to \mathbb{N}$, defined on a regular hexahedral grid of dimensions n_x, n_y, and n_z,

$$G^0 = \{(x_i, y_j, z_k) : 0 \leq i \leq n_x, 0 \leq j \leq n_y, 0 \leq k \leq n_z\},$$

with $x_i = x_0 + i\,h_x, y_j = y_0 + j\,h_y, z_k = z_0 + k\,h_z$. In order to simplify notation, we further assume a consistent grid size $h = h_x = h_y = h_z$. A hierarchy $f^0, f^1, f^2, \ldots$, where each f^l is defined on a grid G^l with grid size $2^l h$ and dimensions $\lceil 2^{-l} n_x \rceil$, $\lceil 2^{-l} n_y \rceil$,$\lceil 2^{-l} n_z \rceil$, can then be computed by iteratively down-sampling the volume data with the factor 2. This process is usually realized by convolving the function f^{l-1} with a suitable filter and then sampling the filtered signal to obtain f^l. We have tested several filters, including box, Gauß and median filter, but found the dilation operator to perform best within the scope of our investigations. This operator selects the largest gray value from the cluster of eight voxels on level $l-1$ that are combined to form the corresponding voxel with double edge length on level l and defines

$$f^l(x_i, y_j, z_k) = \max_{\substack{i' \in \{i, i+1\} \\ j' \in \{j, j+1\} \\ k' \in \{k, k+1\}}} f^{l-1}(x_{i'}, y_{j'}, z_{k'}),$$

where i, j, and k are multiples of 2^l. Given an iso-value v, we can now extract an approximation $M^l(v)$ of the corresponding iso-surface from the down-sampled data set f^l with a standard marching cubes algorithm [17].

Figure 2 shows that the use of other filters tends to tear the thin layer of voxels that represents the object's material apart, hence the topological holes in the extracted iso-surface. In contrast, the dilation operator has a growing effect and for a fixed iso-value, M^l can in fact be proven to encompass the meshes $M^{l-1}, M^{l-2}, \ldots, M^0$ extracted from the finer levels as illustrated in Figure 3. And even if this method may modify the topology of the iso-surface in general, as small holes can vanish as a result of the dilation, it is appropriate for the data sets we consider because they are topologically simple.

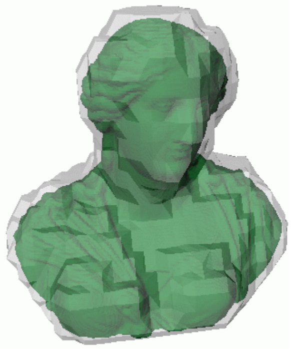

Figure 3: Iso-surfaces $M^3(900)$ (gray) and $M^0(900)$ (green).

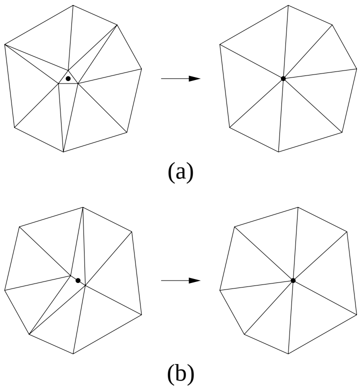

Figure 4: Removing short edges from the extracted iso-surface with a two-pass mesh decimation algorithm.

A typical phenomenon of the marching cubes algorithm is that some of the generated triangles are very small. In fact, whenever a voxel's gray value is close to the specified iso-value v, the algorithm cuts off a corner of the underlying grid with a triangle whose size is the smaller the smaller the difference to v is. Due to their tininess it seems reasonable to assume these triangles not to contain significant geometrical information. And as we finally aim at generating a triangulated iso-surface with evenly distributed vertices, we propose to perform a decimation step before further processing the mesh.

In order to remove all edges that are shorter than a certain threshold length $\alpha 2^l h$ with $\alpha > 0$, we first replace all the triangles with three short edges by a single vertex at their barycentre (see Figure 4.a) and then collapse the remaining short edges to their midpoints (see Figure 4.b). We found $\alpha = 0.5$ to be a good choice and this simple strategy to reduce the number of triangles by approximately 20%.

Due to the growing effect of the dilation operator, the vertices of the base mesh do not lie on the iso-surface at level 0 that we actually want to reconstruct, and we have to shrink the mesh onto that surface as described in the next section. In order to increase robustness and performance of that algorithm, we utilize the previously constructed hierarchy of volumes by iteratively fitting the mesh to the next coarser level. We first move the vertices to the iso-surface at level $l-1$, then to the one at level $l-2$, and so on, until we finally arrive at level 0. Note that this always guarantees the vertices of the current mesh to be close to the iso-surface, namely within distance of 2 voxels. This helps to avoid self-intersections of the triangulation after projecting the vertices which may occur if the distance is too large as mentioned in [12].

4 Iso-Surface Fitting

The essential step of our hierarchical iso-surface extraction algorithm is to adaptively fit the current mesh to the iso-surface of the volume at a certain level l. Such an iso-surface $S^l(v)$ is defined as

$$S^l(v) = \{(x, y, z) : \tilde{f}^l(x, y, z) = v\}$$

where $\tilde{f}^l : [G^l] \to \mathbb{R}$ is the continuous extension of f^l which trilinearly interpolates the values $f^l(G^l)$. The three ingredients of this fitting procedure which are repeated iteratively are the following:

1. Moving the vertices to the iso-surface (projection).
2. Improving the distribution of the vertices (relaxation).
3. Adaptively subdividing the mesh (refinement).

4.1 Projection

As the iso-surfaces S^{l+1} and S^l are different, the vertices of the current mesh, which is in fact an approximation of S^{l+1}, will not lie on S^l and we need a method for projecting them onto that surface. In principle, this can be done by finding the first intersection of a ray emanating from that vertex in a certain direction with S^l, but the question remains how to determine the direction of that ray.

We could, for example, use the gradient of the gray value function f^0, as it is often done in volume rendering [16, 28, 21, 11], but for our kind of data this choice is inappropriate for the following reason. Since the objects that we wish to reconstruct are made of a rather homogeneous material, the volume data from a CT scan with infinitesimal resolution would ideally be a binary data set with gray value zero in those voxels which represent the air surrounding the object and a material-dependent constant gray value in all other voxels, so that the gradient of the gray value function is either zero or undefined. In practice, however, a CT scanner has finite resolution only and we also have to deal with noise, but that certainly does not improve the quality of the gray value gradient.

Another choice that has also been proposed in [9, 8] is the gradient of the distance function $d^l : [G^l] \to \mathbb{R}$ which gives the shortest signed distance of a point to the iso-surface S^l. For volumes, such a distance function is usually defined by the values at the grid points G^l and trilinear interpolation, just as $\tilde{f}^l$, and the values at the grid points are determined by a fast marching method [23]. The distance gradient proved to be a better choice than the gray value gradient but it also has some potential drawbacks. Firstly, it is not properly defined everywhere, because it is discontinuous along the medial axis of S^l in theory, and the evaluation of the gradient is extremely unstable near the medial axis for that reason. However, since the current mesh is guaranteed to be close to S^l this is not an issue for our computations, but a more serious drawback is that the distance gradient is not capable of moving vertices inside a concave region of the iso-surface as illustrated in Figure 5.

We therefore propose to move the vertex $\mathbf{v}$ along the direction of the normal vector $\mathbf{n_v}$ at $\mathbf{v}$ which can either be found by averaging the normals of the triangles adjacent to $\mathbf{v}$ or, as we did, by normalizing the curvature normal vector [4]

$$\boldsymbol{\kappa}(\mathbf{v}) = \sum_{\mathbf{w} \in N_\mathbf{v}} (\cot \alpha_\mathbf{w} + \cot \beta_\mathbf{w})(\mathbf{v} - \mathbf{w}),$$

where $N_\mathbf{v}$ is the set of $\mathbf{v}$'s neighboring vertices and $\alpha_\mathbf{w}$ and $\beta_\mathbf{w}$ are the angles opposite to the edge $\overline{\mathbf{vw}}$ in the adjacent triangles.

Once the normal is computed, it remains to decide whether the intersection with the iso-surface can be found in the positive or negative direction. Figure 6 shows which different cases occur and we

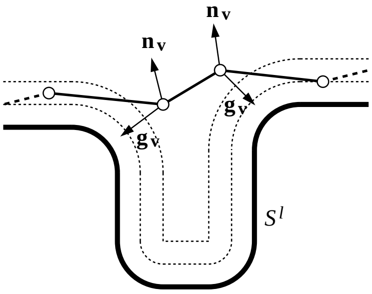

Figure 5: Iso-surface S^l with iso-distance lines (dotted) and distance gradients $\mathbf{g}_\mathbf{v}$ and normals $\mathbf{n}_\mathbf{v}$ at two vertices of the mesh.

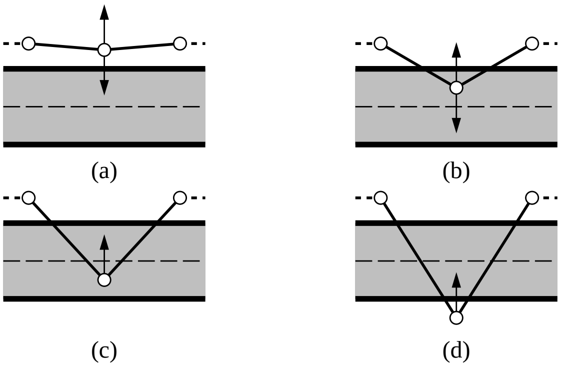

Figure 6: Distance gradient and normal vector at a vertex in four different situations. The gray-shaded region indicates the region enclosed by the iso-surface and the dashed line its medial axis.

use the distance function and its gradient to recognize them. Normally, the vertex is near the iso-surface and we can simply decide on the sign of the distance function on which side of the surface it lies. If $d^l(\mathbf{v}) > 0$ then the vertex lies 'outside', so that we move it into the opposite direction of its normal vector (a) and if $d^l(\mathbf{v}) < 0$ then it is located 'inside' and needs to be pushed along the normal direction (b). Both cases have in common that the normal vector $\mathbf{n}_\mathbf{v}$ and the distance gradient $\mathbf{g}_\mathbf{v}$ point in opposite directions. But it may also happen that $\mathbf{n}_\mathbf{v}$ and $\mathbf{g}_\mathbf{v}$ are oriented similarly as in (c) and (d), indicating that the vertex is beyond the medial axis and close to the 'wrong' iso-surface. This can happen if the refinement step insert new vertices into the mesh at highly curved regions where the object is also very thin so that the two parts of the iso-surface are close. In this case we consider the ray along the positive normal direction and find the first (c) or second (d) intersection with the iso-surface. In order to distinguish between the cases (a,b) and (c,d) we determine the sign of the scalar product $<\mathbf{n}_\mathbf{v}|\mathbf{g}_\mathbf{v}>$. Although the distance gradient is not defined on the medial axis no problems will occur, because in both cases (b) and (c) the vertex will be moved along the positive normal direction.

4.2 Relaxation

A problem of this projection method is that it may lead to a local clustering of vertices or even self-intersections of the triangulation. But by additionally applying a relaxing force we can overcome this drawback and ensure an even distribution of the vertices over the iso-surface. A common approach is to apply a discrete version of the Laplacian,

$$L(\mathbf{v}) = \frac{1}{|N_\mathbf{v}|} \sum_{\mathbf{w} \in N_\mathbf{v}} (\mathbf{w} - \mathbf{v}),$$

as it was done, for example, in [24, 4, 12] in order to smooth or denoise meshes. But this operator has a shrinking effect on the mesh and moves vertices far off the iso-surface in highly curved regions. We therefore follow the strategy in [30] and use only the tangential part of the Laplacian,

$$T(\mathbf{v}) = L(\mathbf{v}) - <L(\mathbf{v})|\mathbf{n}_\mathbf{v}> \mathbf{n}_\mathbf{v},$$

for smoothing the parameterization of the mesh and keeping the vertices close to the iso-surface.

4.3 Refinement

In order to approximate the iso-surface within a user-specified accuracy and to capture local detail we adaptively subdivide a triangle of the mesh depending on a refinement criterion. For a triangle $T = [\mathbf{u}, \mathbf{v}, \mathbf{w}]$ we evaluate the distance function at a number of sample points $\alpha\mathbf{u} + \beta\mathbf{v} + \gamma\mathbf{w}$ with $\alpha + \beta + \gamma = 1$, and quadrisect the triangle if at least one of these vertices is further from the iso-surface than a given ε. After subdividing a triangle, the newly inserted vertices are projected onto the iso-surface as described in Section 4.1.

There are a few restrictions in this adaptive subdivision approach. As we want to keep the number of special configurations small we only allow balanced meshes, i.e. the refinement level of two neighboring triangles may only differ by one and we use the special technique of red-green triangulations [1, 25, 26] to avoid cracks in the mesh where two triangles from different levels meet.

5 Results

We shall now present some examples which show the capability of our algorithm in the context of archaeological surface reconstruction. The leftmost picture in Figure 1 shows the base mesh as extracted from the coarsest volume after decimation. In the middle, the meshes are shown after fitting them to the iso-surfaces of the next finer volume levels. In both steps a uniform subdivision step was performed. The rightmost picture shows the final result after two further adaptive subdivision steps which was fitted to the iso-surface at the finest volume resolution and smoothed within the user-specified tolerance.

As can be seen from the vertical section of that mesh in Figure 7, we did not only reconstruct the outer but also the inner surface of this item which appears to be hollow, indeed. Such cross-sections

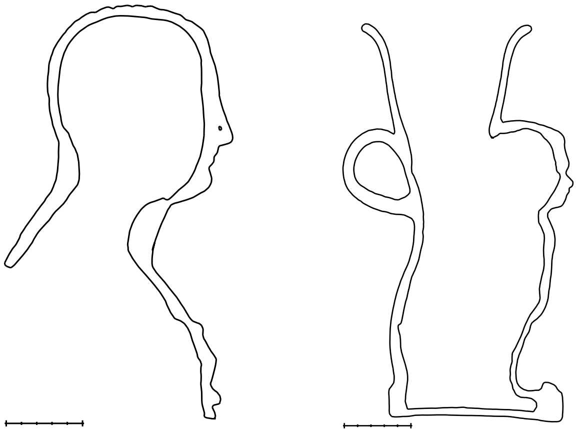

Figure 7: Examples of vertical sections from the reconstructed surfaces in Figures 1 and 8.

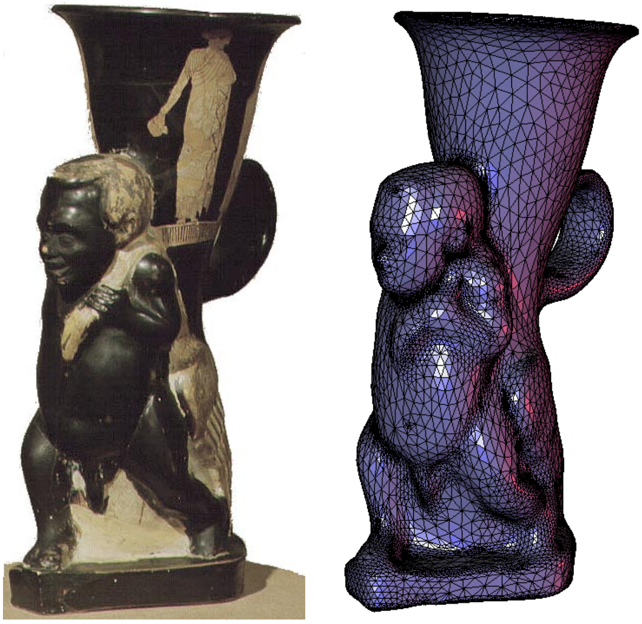

Figure 8: Original and reconstruction of a drinking pot featuring a pygmy (Athens, 5th century b.c.).

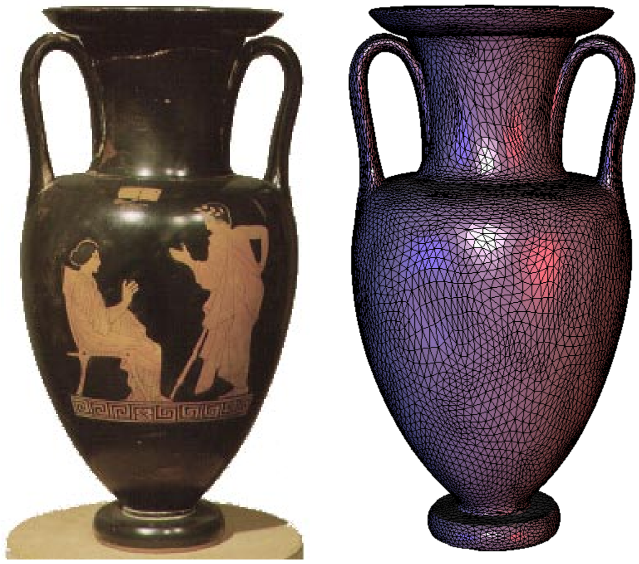

Figure 9: Original and reconstruction of a red-figured amphora (Athens, 5th century b.c).

are of vital importance in the daily life of an archaeologist since they allow to study the profile and the wall thickness of the object which helps to identify the period in which they were crafted or even the specific potter. The other section in Figure 7 was taken from the surface shown in Figure 8 and a final example is illustrated in Figure 9.

In Table 1 we have listed the size of the volumes which were used to extract the models, the size of the final triangle meshes, and the time which was required to reconstruct the mesh. The times were measured on an AMD Athlon with 1.2 GHz. Also listed is the approximation error of the mesh compared to a mesh produced by using marching cubes on the full resolution of the volume data set.

name	volume	triangles	approx. error	time (s)
isis	256x256x128	110916	0.032%	16.05
amphora	256x256x256	94608	0.018%	79.02
pygmy	256x256x256	119292	0.036%	87.5

Table 1: Size, approximation error and reconstruction time of the presented models.

6 Conclusion

In this paper we presented a new approach for hierarchically extracting iso-surfaces as semi-regular meshes from volume data. Our method utilizes a multiresolution representation of the given volume and creates a coarse base mesh by extracting and post processing the iso-surface from the coarsest resolution. Then, the iso-surfaces on the finer volume levels are iteratively captured by a fitting procedure that not only projects the vertices of the current mesh onto the iso-surface but also takes care of the vertex distribution and adaptively refines the mesh to approximate local detail.

The method was inspired by the shrink-wrapping approach for remeshing arbitrarily connected triangle meshes [12] but our hierarchical setting improves robustness and performance of that algorithm. A potential drawback of the method is that the dilation operator may change the topology of the iso-surface as small holes can disappear. Though this is not an issue for the data sets we considered, since they were topologically simple, this remains a problem to be addressed in future work.

7 Acknowledgments

We would like to thank Dr. Martin Boss from the Archeological Institute from the University of Erlangen-Nuremberg for providing the archeological items and Dr. Bernd Tomandl from the Neuroradiology for scanning the objects.

References

[1] R. E. Bank, A. H. Sherman, and A. Weiser. Refinement algorithms and data structures for regular local mesh refinement. In R. Stepleman, editor, *Scientific Computing*, pages 3–17, Amsterdam, 1983. IMACS/North Holland.

[2] M. Bertram, M. A. Duchaineau, B. Hamann, and K. I. Joy. Bicubic subdivision-surface wavelets for large-scale isosurface representation and visualization. In *Visualization 2000 Proceedings*, pages 389–396, 2000.

[3] A. Certain, J. Popović, T. DeRose, T. Duchamp, D. Salesin, and W. Stuetzle. Interactive multiresolution surface viewing. In *ACM Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 91–98, 1996.

[4] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *ACM Computer Graphics (SIGGRAPH '99 Proceedings)*, pages 317–324, 1999.

[5] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 173–182, 1995.

[6] T. Gerstner and R. Pajarola. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In *IEEE Visualization 2000*, pages 259–266, 2000.

[7] H. Hoppe. Progressive meshes. In *ACM Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 99–108, 1996.

[8] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction. In *ACM Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 295–302, 1994.

[9] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *ACM Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 19–26, 1993.

[10] A. Khodakovsky, W. Sweldens, and P. Schröder. Progressive geometry compression. In *ACM Computer Graphics (SIGGRAPH '00 Proceedings)*, pages 271–278, 2000.

[11] J. Kniss, G. Kindlmann, and C. Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Visualization 2001 Proceedings*, 2001.

[12] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. In *Computer Graphics Forum (EUROGRAPHICS '99 Proceedings)*, volume 18, pages 119–130, 1999.

[13] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *ACM Computer Graphics (SIGGRAPH '01 Proceedings)*, pages 57–66, 2001.

[14] U. Labsik, L. Kobbelt, R. Schneider, and H.-P. Seidel. Progressive transmission of subdivision surfaces. *Computational Geometry*, 15:25–39, 2000.

[15] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin. MAPS: Multiresolution adaptive parameterization of surfaces. In *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 95–104, 1998.

[16] M. Levoy. Display of surfaces from volume data. *IEEE Comp. Graph. & Appl.*, 8:29–37, 1988.

[17] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *ACM Computer Graphics (SIGGRAPH '87 Proceedings)*, pages 163–169, 1987.

[18] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics*, 16:34–73, 1997.

[19] C. Montani, R. Scateni, and R. Scopigno. A modified lookup table for implicit disambiguation of marching cubes. *The Visual Computer*, 10:353–355, 1994.

[20] G. Nielson and B. Hamann. The asymptotic decider: resolving the ambiguity in marching cubes. In *Visualization 1991 Proceedings*, pages 83–91, 1991.

[21] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl. Interactive volume rendering on standard PC graphics hardware using multi-textures and multi-stage rasterization. In *Proc. SIGGRAPH/Eurographics Workshop on Graphics Hardware*, 2000.

[22] P. Schröder and W. Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 161–172, 1995.

[23] J. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, UK, 1999.

[24] G. Taubin. A signal processing approach to fair surface design. In *ACM Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 351–358, 1995.

[25] M. Vasilescu and D. Terzopoulus. Adaptive meshes and shells: Irregular triangulation, discontinuities, and hierarchical subdivision. In *Proceedings of Computer Vision and Pattern Recognition conference*, pages 829–832, 1992.

[26] R. Verfürth. *A review of a posteriori error estimation and adaptive mesh refinement techniques*. Wiley-Teubner, 1996.

[27] G. Weber, O. Kreylos, T. Ligocki, H. Hagen, B. Hamann, and K. Joy. Extraction of crack-free isosurfaces from adaptive mesh refinement data. In *Proceedings of VisSym 2001*, pages 25–34, 2001.

[28] R. Westermann and T. Ertl. Efficiently using graphics hardware in volume rendering applications. In *ACM Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 169–179, 1998.

[29] R. Westermann, L. Kobbelt, and T. Ertl. Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Visual Computer*, 15(2):100–111, 1999.

[30] Z. Wood, M. Desbrun, P. Schröder, and D. Breen. Semi-regular mesh extraction from volumes. In *Visualization 2000 Proceedings*, pages 275–282, 2000.

[31] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. In *ACM Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 259–268, 1997.